BUILDING YOUR LOW-COST IOT PLATFORM USING WAZIUP DEMO KIT

Dr. Mamour DIOP

Tekki48 IoT & Big Data WAZIHack
December 14th, 2016

CONTENT


WAZIUP

IOT DOMAINS IN AFRICA

Irrigation & Agriculture

Livestock farming

Fish farming & Aquaculture

Storage & logistic

Health

Water quality

RURAL SENSING

Moisture/Temperature of storage areas

10-15kms

GSM GPRS

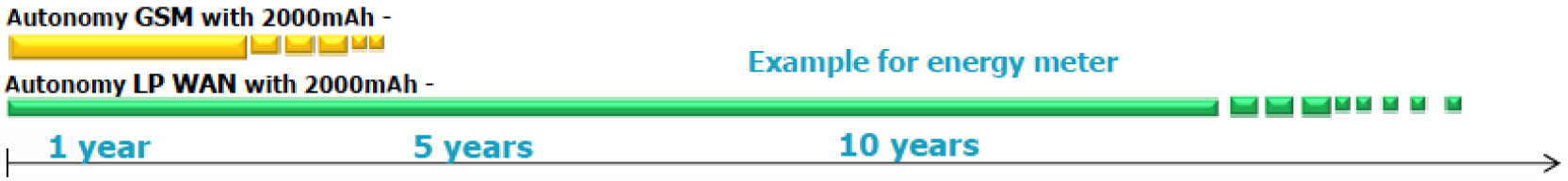
Pay subscription
Limitation of coverage
High energy consumption

Technology	2G	3G	LAN
Range (I=Indoor, O=Outdoor)	N/A	N/A	O: 300m I: 30m
Tx current consumption	200-500mA	500-1000mA	100-300mA
Standby current	2.3mA	3.5mA	NC

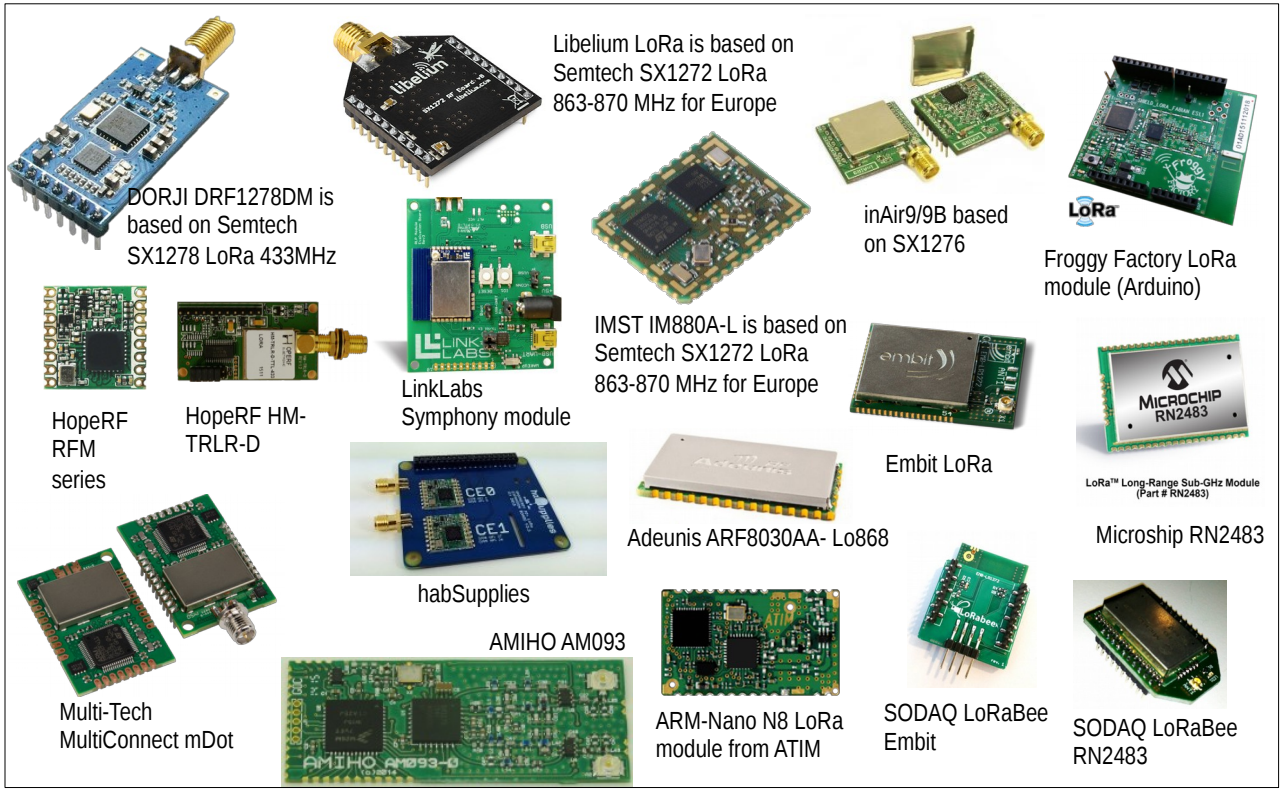
LOW POWER WAN (LPWAN) ?

Tables from Semtech

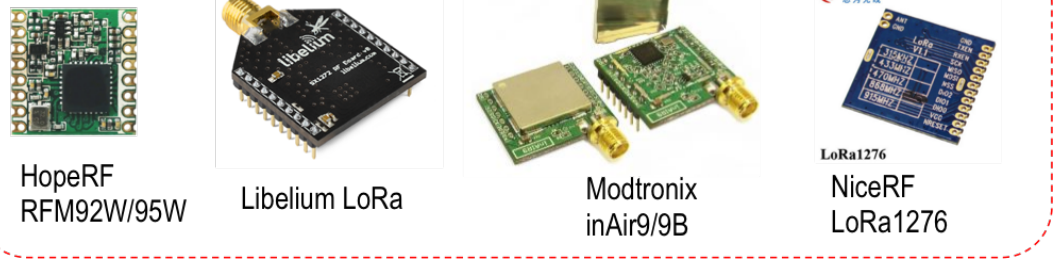
Technology	2G	3G	LAN	ZigBee	Lo Power WAN
Range (I=Indoor, O=Outdoor)	N/A	N/A	O: 300m I: 30m	O: 90m I: 30m	Same as 2G/3G
Tx current consumption	200-500mA	500-1000mA	100-300mA	18mA	18mA
Standby current	2.3mA	3.5mA	NC	0.003mA	0.001mA
Energy harvesting (solar, other)	No	No	No	Possible	Possible
Battery 2000mAh (LR6 battery)	4-8 hours(com) 36 days(idle)	2-4 hours(com) X hours(idle)	50 hours(com) X hours(idle)	60hours (com)	120 hours(com) 10 year(idle)
Module Revenue Annually	12 $	20 $	4 $	$3	3 $



LORA MODULES FROM SEMTECH'S SX127X CHIPS



LoRa radios that our library already supports




WAZIUP

LOW-COST HARDWARE

WHAT IS ARDUINO?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.

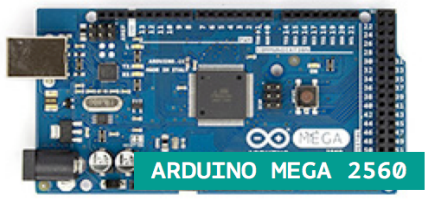
ARDUINO BOARD

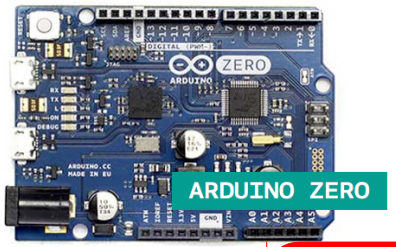
Arduino senses the environment by receiving inputs from many sensors, and affects its surroundings by controlling lights, motors, and other actuators.

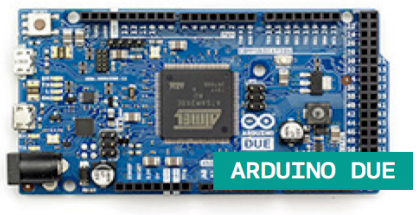
ARDUINO SOFTWARE

You can tell your Arduino what to do by writing code in the Arduino programming language and using the Arduino development environment.

```
void setup() {
}

void loop() {
}
```

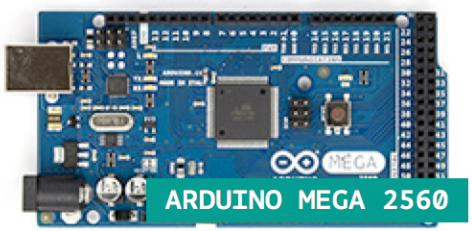


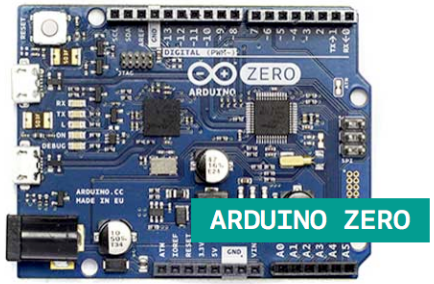


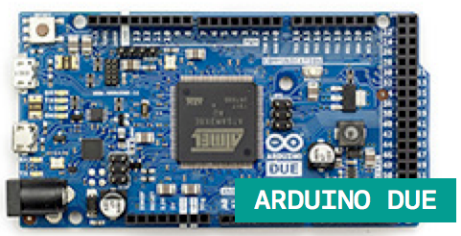


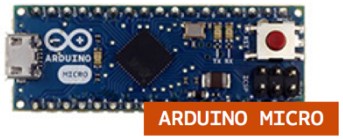
HARDWARE/SOFTWARE BUILDING BLOCKS INTEGRATION


ARDUINO UNO


ARDUINO MEGA 2560

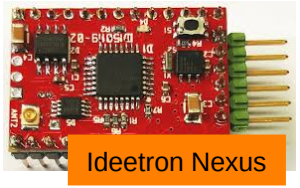

ARDUINO ZERO


ARDUINO DUE


ARDUINO MICRO

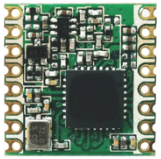

ARDUINO PRO MINI

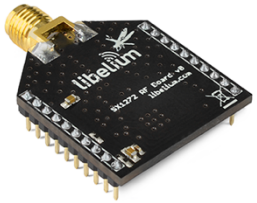

ARDUINO NANO


Ideetron Nexus

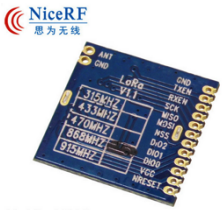

Teensy3.1/3.2

LoRa radios that our library already supports


HopeRF RFM92W/95W


Libelium LoRa

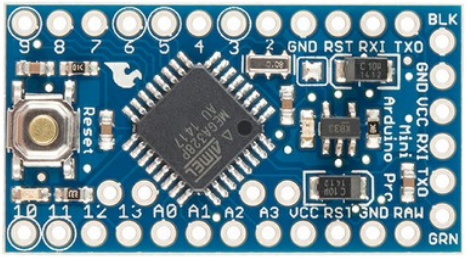

Modtronix inAir9/9B

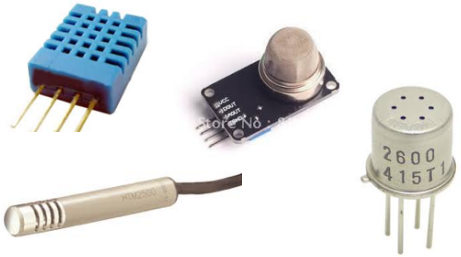

LoRa1276
NiceRF LoRa1276

Long-Range communication library

GENERIC SENSOR IOT DEVICE

- Build low-cost, low-power, Long-range enabled generic platform
- Methodology for low-cost platform design
- Technology transfers to user communities, economic actors, stakeholders,…





READY-TO-USE TEMPLATES

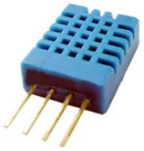
Physical sensor reading

Physical sensor reading

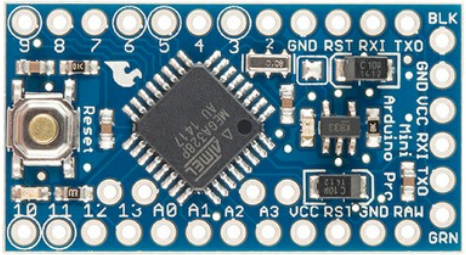
Physical sensor reading

IoT catalogue

Code generation



Physical sensor management



Activity duty-cycle, low power

Security

Long-range transmission

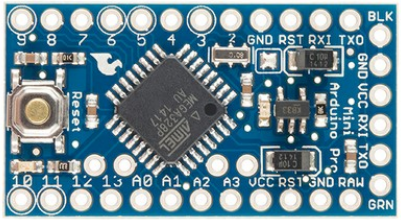
Logical sensor management

GENERALIZATION

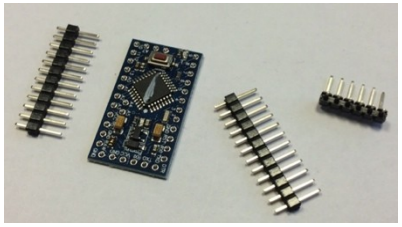
- Depending on the sensor type, getting the physical measure from the analog/digital value follows a specific function provided by the sensor's manufacturer
- Depending on the microcontroller board, the number of I/O pins and the operating voltage may differ
- However the process is always the same:
  - Coonect the sensor to the microcontroller board
  - Read analog or digital pin
  - Convert read value into meaningful physical measure
  - Then process and/or transmit

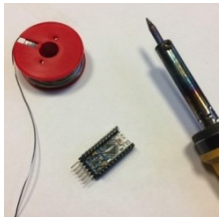
EASY INTEGRATION AND CUSTOMIZATION

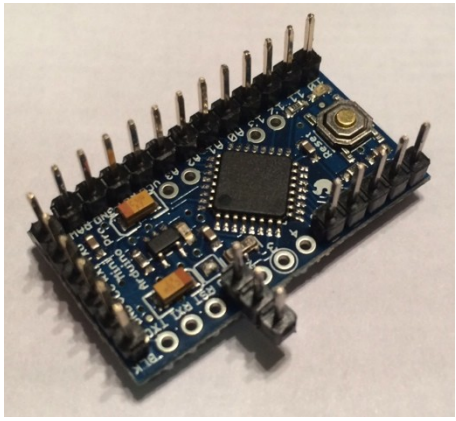
Arduino Pro Mini

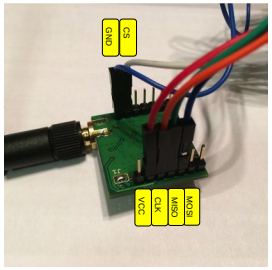


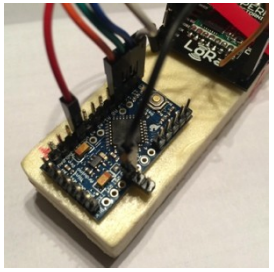
3.3v and 8MHz version

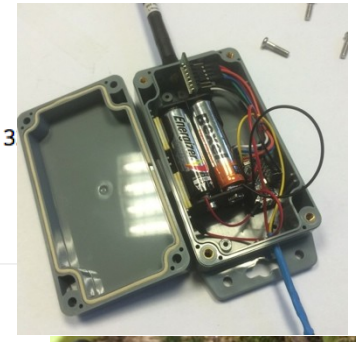


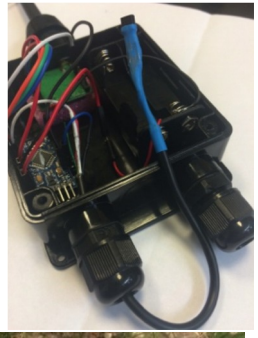












Avec la bootloader 1 pcs Pro Mini ATMEGA328 Pro Mini 3 MHz pour Arduino

View original title in English

4.9 (417 Votes) | 434 Commandes

Prix : € 1,49 / Kit

Trouvez plus de deals sur l'App

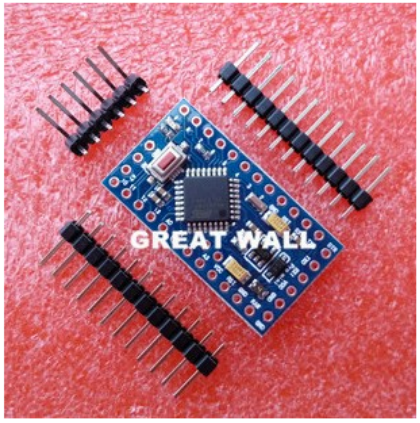
Livraison : € 0,29 vers France via China Post Ordinary Small Pack

Livraison : 15-34 jours (envoyé en 7 jours ouvrables)

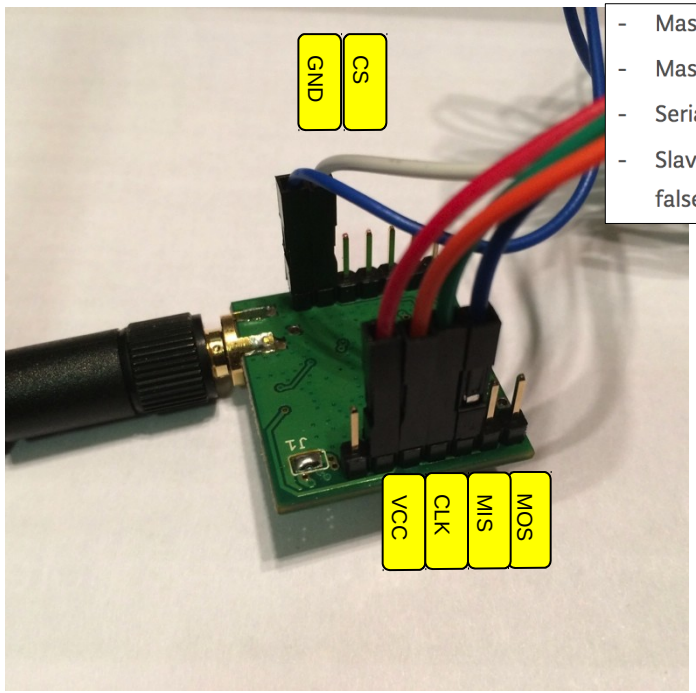
Quantité : – 1 + Kit (55350 Kits available)

Montant total : € 1,78

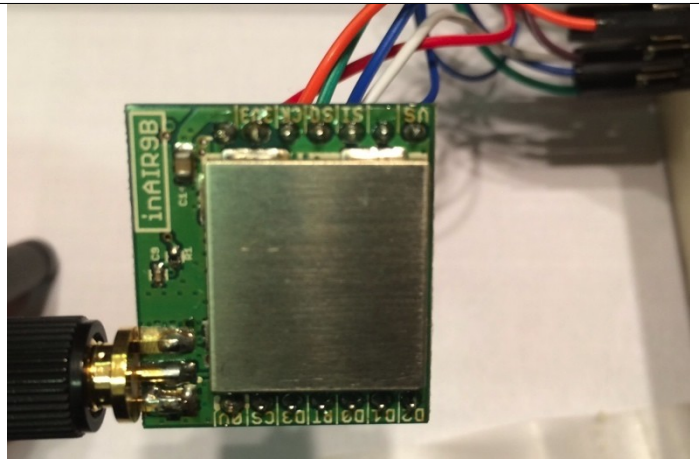
Acheter maintenant | Ajouter au panier



THE RADIO MODULE

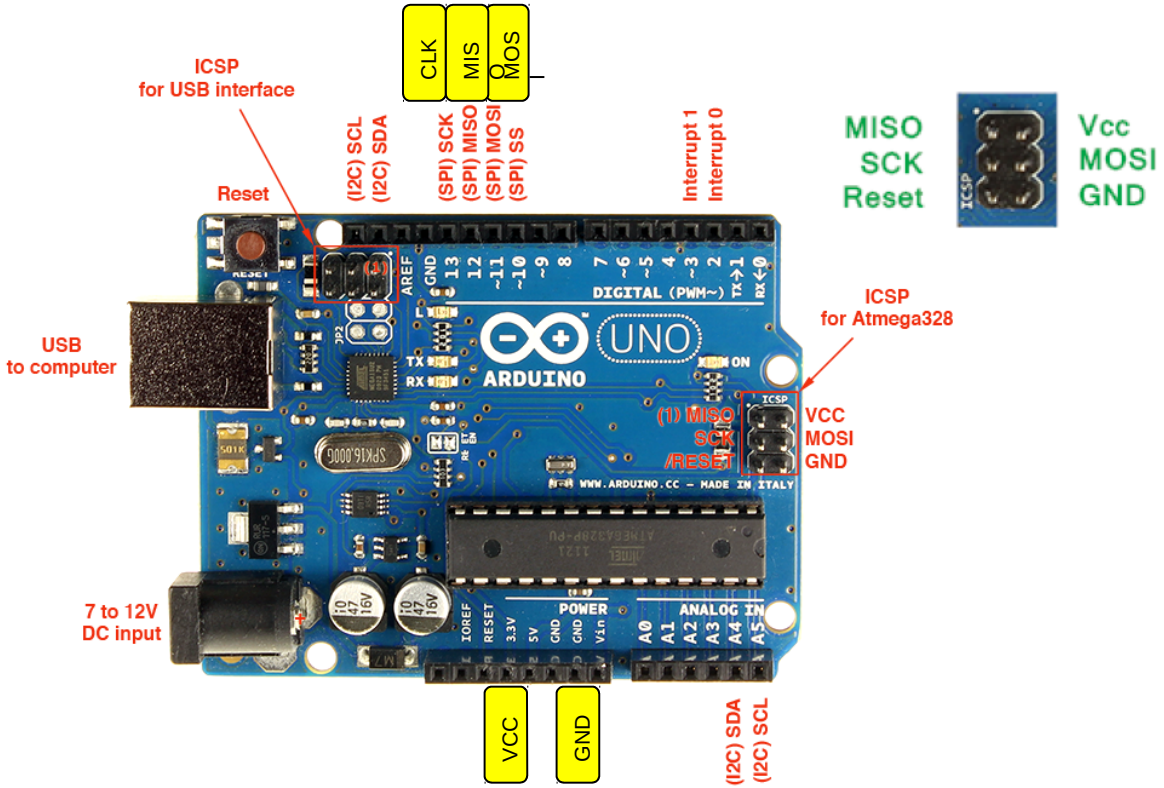


- Master In Slave Out (MISO) - The Slave line for sending data to the master,
- Master Out Slave In (MOSI) - The Master line for sending data to the peripherals,
- Serial Clock (SCK) - The clock pulses which synchronize data transmission generated by the master, and
- Slave Select pin - allocated on each device which the master can use to enable and disable specific devices and avoid false transmissions due to line noise.



If you go for the inAir9 from Modtronix, then the header pins can come fully assembled. Order with the 6mm header pins to have enough length to connect F/F breadboard cables (left). Connect the SPI pins with the F/F cables. Try to use different colors. I use the following colors: MOSI (blue), MISO (green), CS (white), CLK (orange). Then connect also the VCC (red) and the GND (black or any other dark color) of the radio board.

CONNECTING THE RADIO MODULE

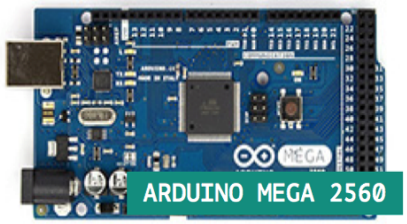


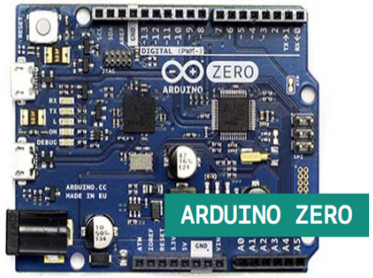
SPI: 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Arduino MEGA and the old Duemilanove and Diecimila Arduino boards

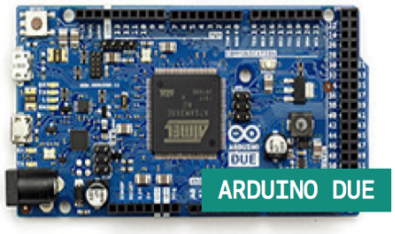
CONNECTING THE SPI CS

- The SPI Chip Select (CS) or SS (Slave Select) of the radio module needs to be connected to a specific pin on the board.

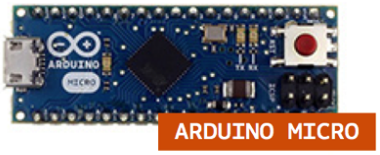
- For boards with UNO format, CS pin 2





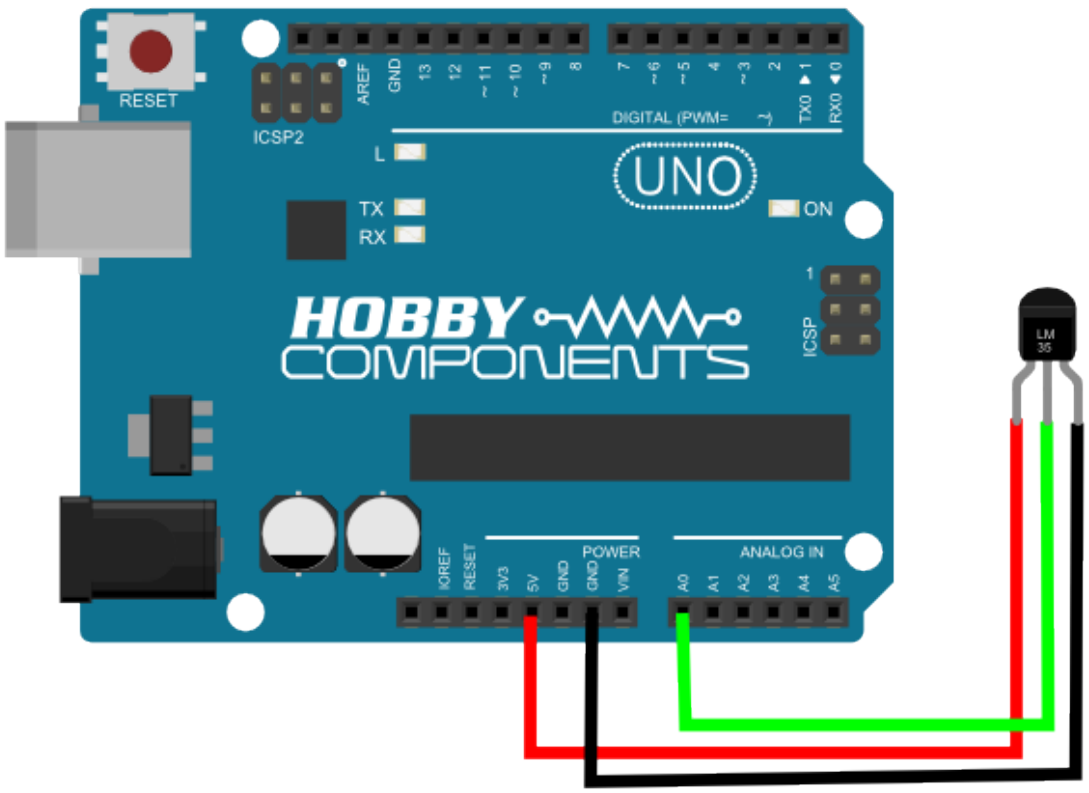


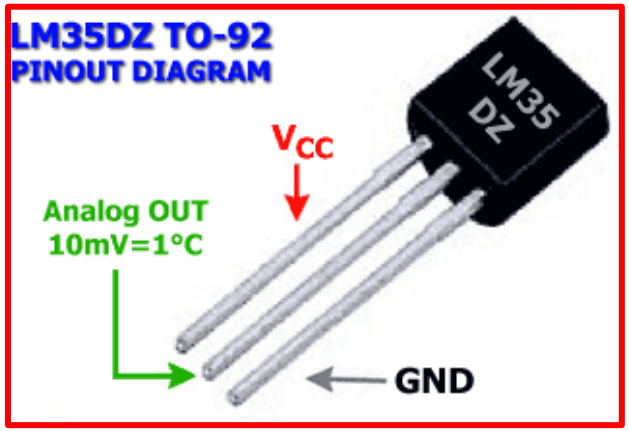
- For boards with UNO format, CS pin 10



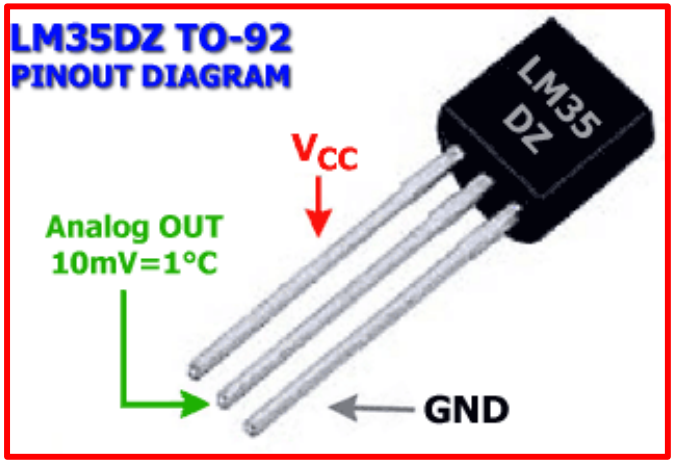
CONNECTING A SENSOR

Arduino Uno





UNDERSTANDING ANALOG OUTPUT

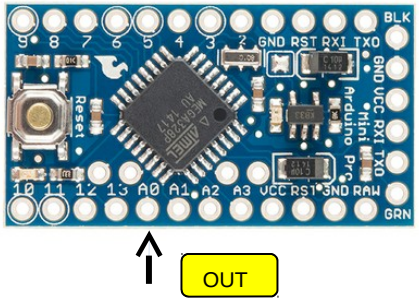


Vcc is 3.3V (the output of digital 8 to power the sensor)

If 0 means 0V and 1024 means 3300mV (10-bit resolution) then 3300mV/1024=3.22mV is the granularity of the measure

A digital value of 100 means 100*3.22mV=322mV
If the sensor output is 10mV/1°C then the physical temperature is 322mV/10mV=32.2°C

READING ANALOG PIN VALUE

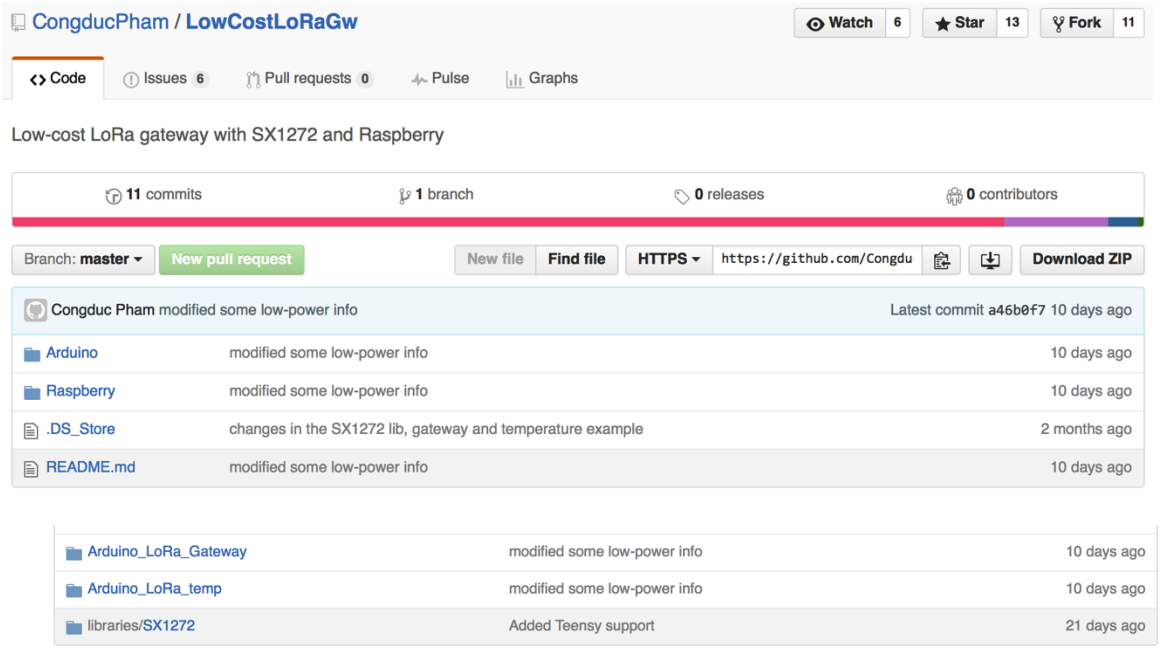



```
// sensor output connected to A0 analog pin

value = analogRead(A0);

// now need to convert to Celcius degree
```

And converting into celcius

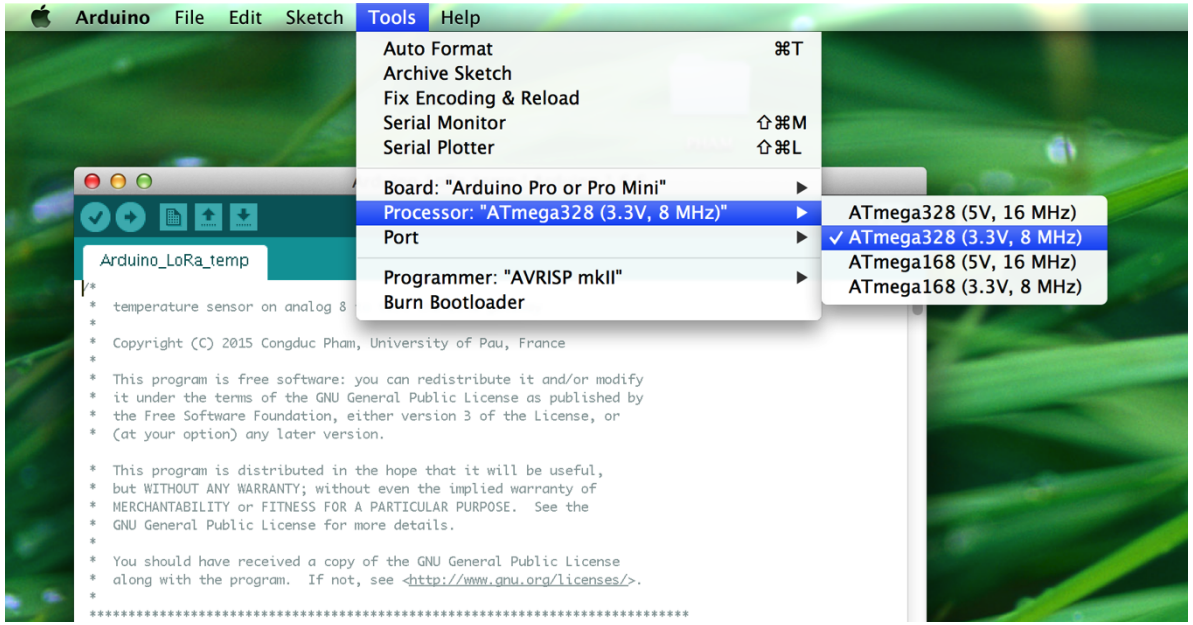
```
Temp = value * 3300.0/1024.0; // 3300/1024=3.22

Temp = Temp / 10;        // 10mV means 1°C

// now process and transmit the data
```

GETTING THE SOFTWARE



Fisrt, you will need the Arduino IDE 1.6.6 or later (left). Then get the LoRa library from our github: https://github.com/CongducPham/LowCostLoRaGw (right).
Go into the Arduino folder and get both Arduino_LoRa_temp and SX1272 folder. Copy Arduino_LoRa_Simple_temp into your “sketch” folder and SX1272 into “sketch/libraries”

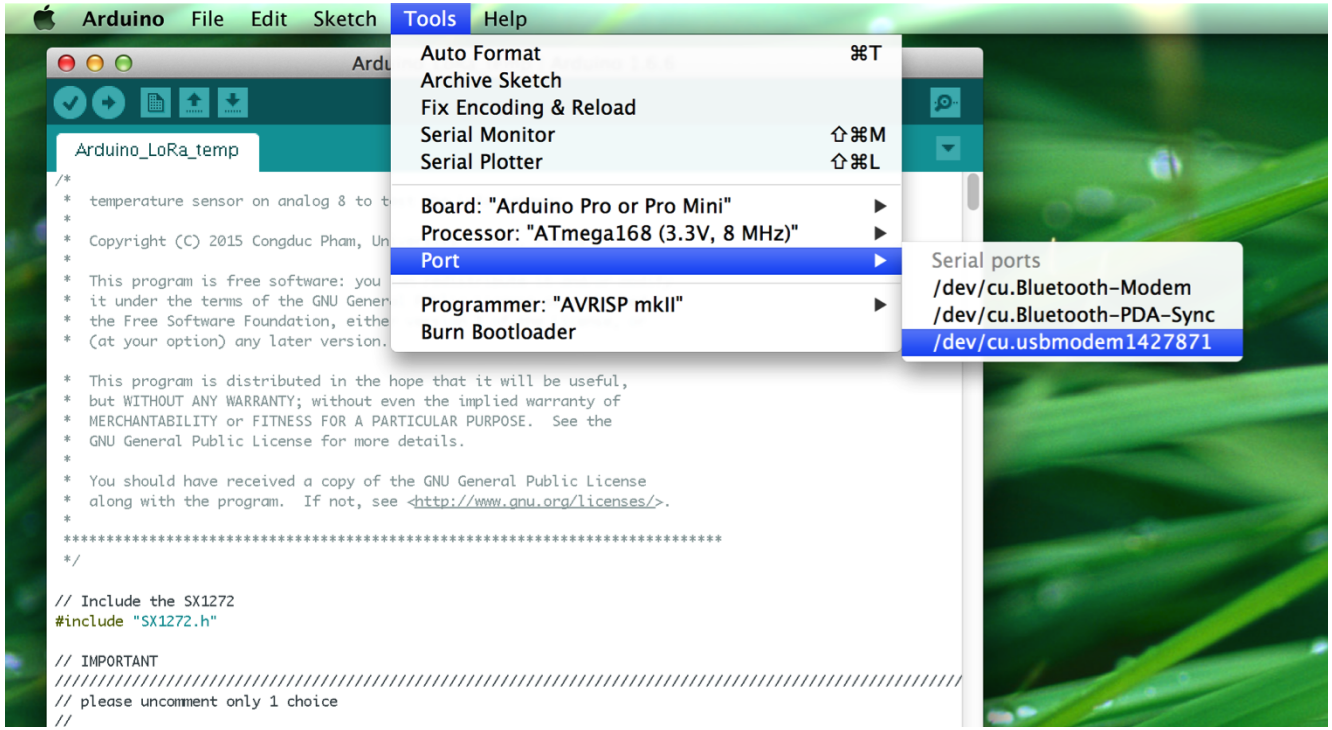
COMPILING



Open the Arduino_LoRa_temp sketch and select the "Arduino/Genuino Uno" board.

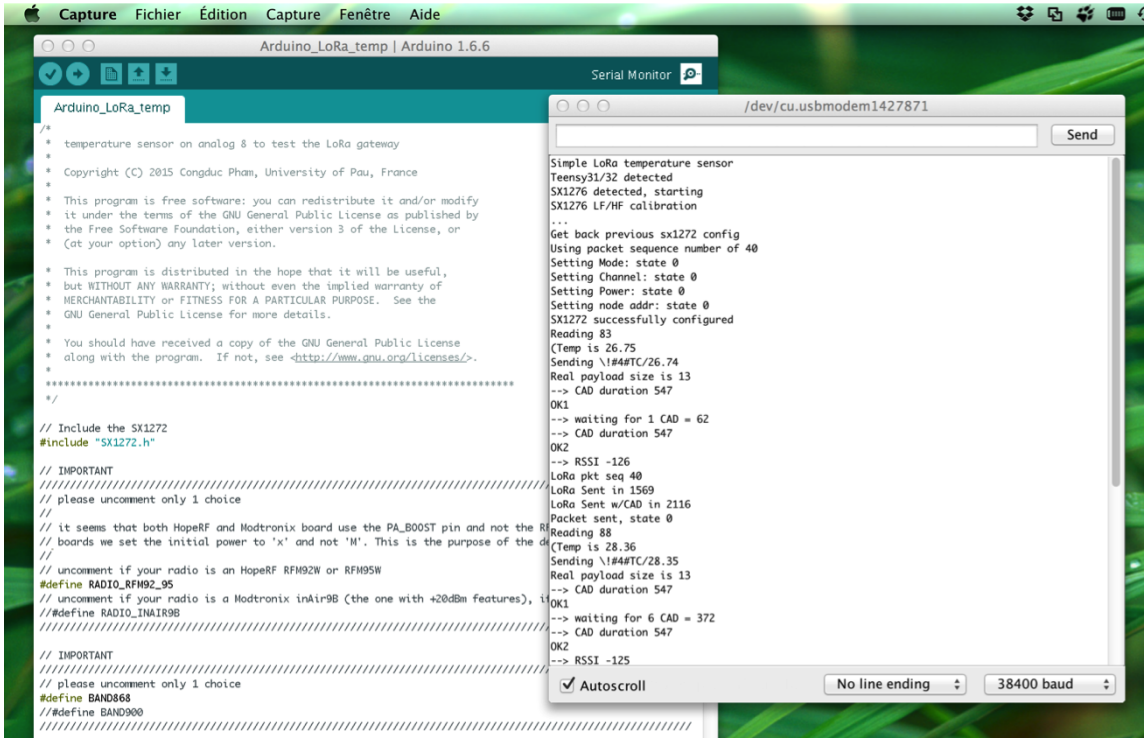
Then, click on the « verify » button 

UPLOADING



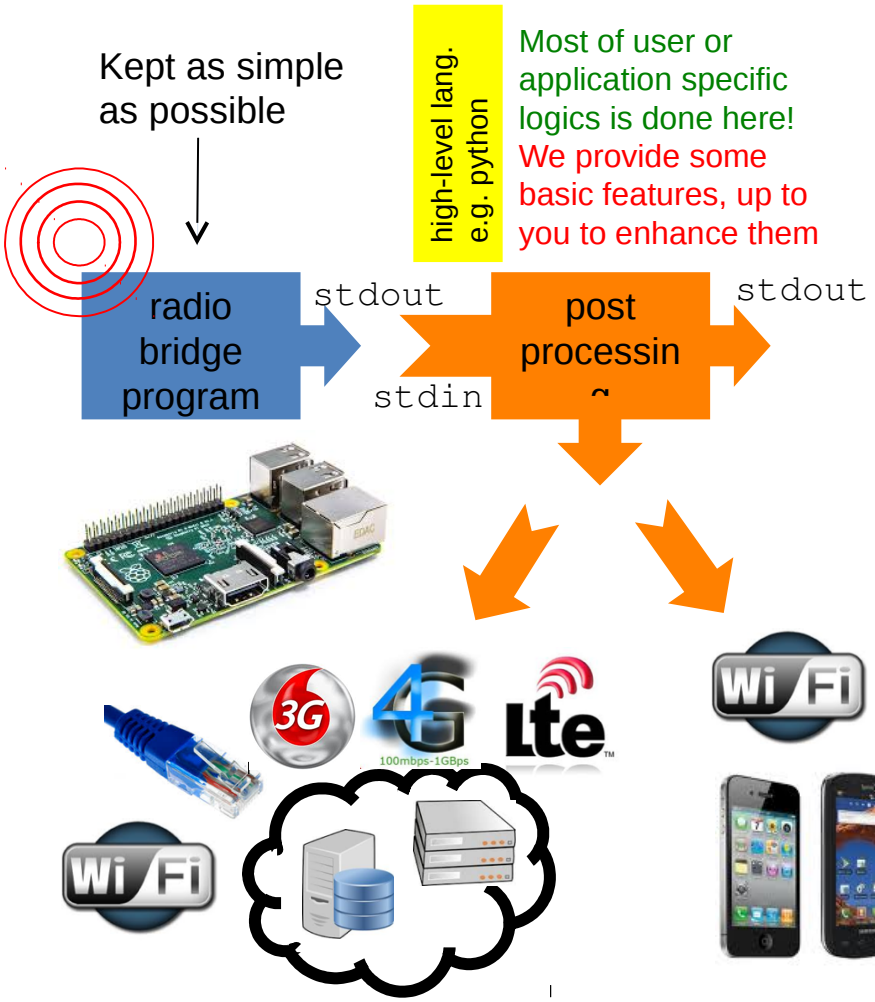
Connect the USB end to your computer and the USB port should be detected in the Arduino IDE. Select the serial port for your device. It may have another name than what is shown in the example. Then click on the « upload » button

SERIAL MONITOR



You can see the output from the sensor if it is connected to your computer. Use the Arduino IDE « serial monitor » to get such output, just to verify that the sensor is running fine, or to debug new code. Be sure to use 38400 baud.

OUR LOW-COST GATEWAY ARCHITECTURE



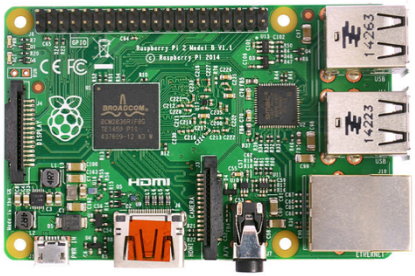
lora_gateway program

Modified SX1272 lib

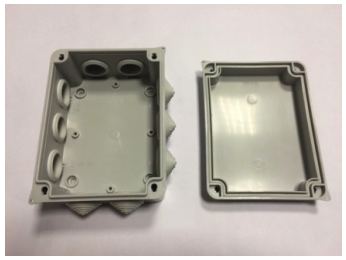
ArduPi lib

Raspbian

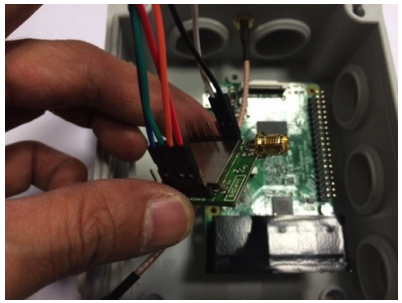
GET THE RASPBERRY



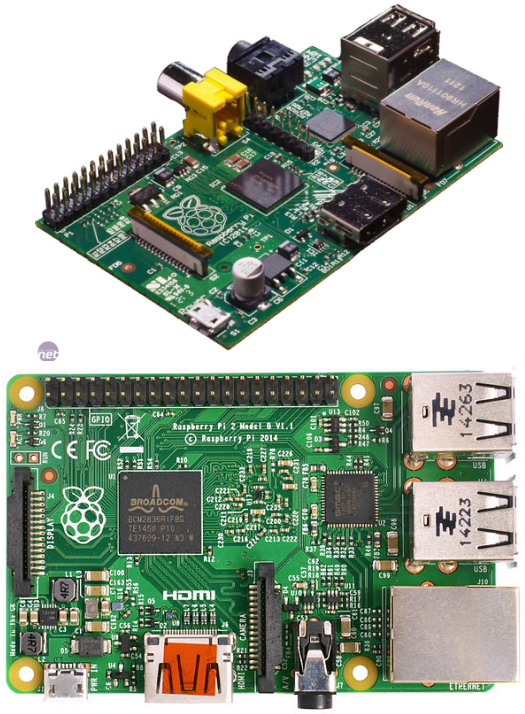
We can use all model of Raspberry. The most important usefull feature is the Ethernet interface for easy Internet connection. Then WiFi and Bluetooth can be added with USB dongles. **RPI3 provides built-in Ethernet, WiFi and Bluetooth**!

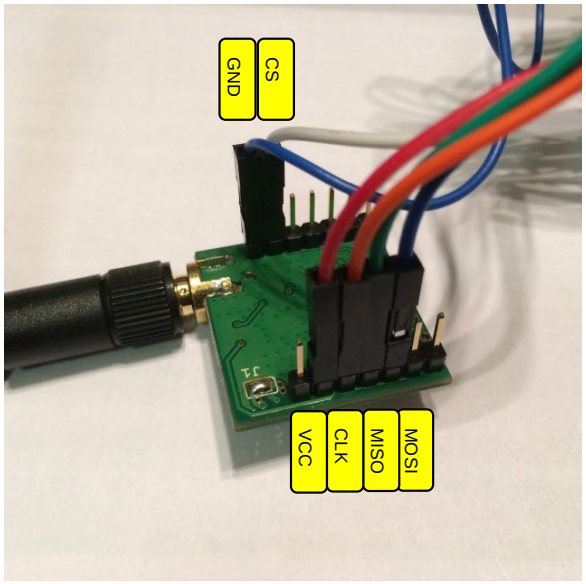


Less than 50€



CONNECTING THE RADIO MODULE (1)

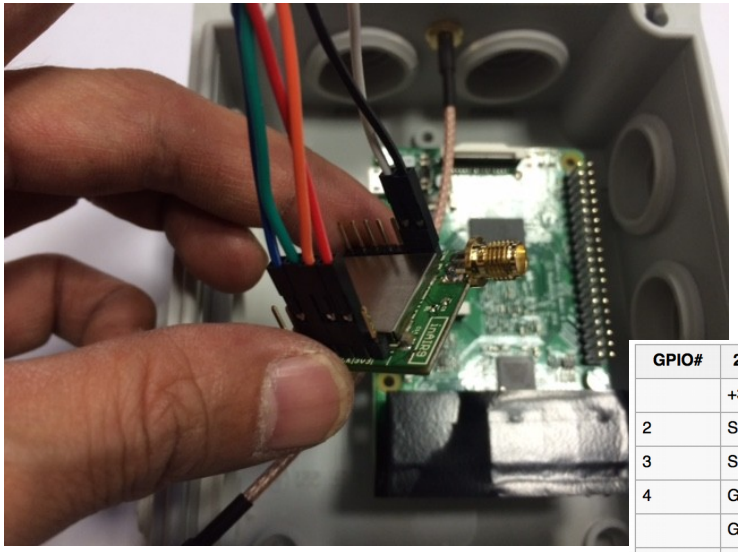


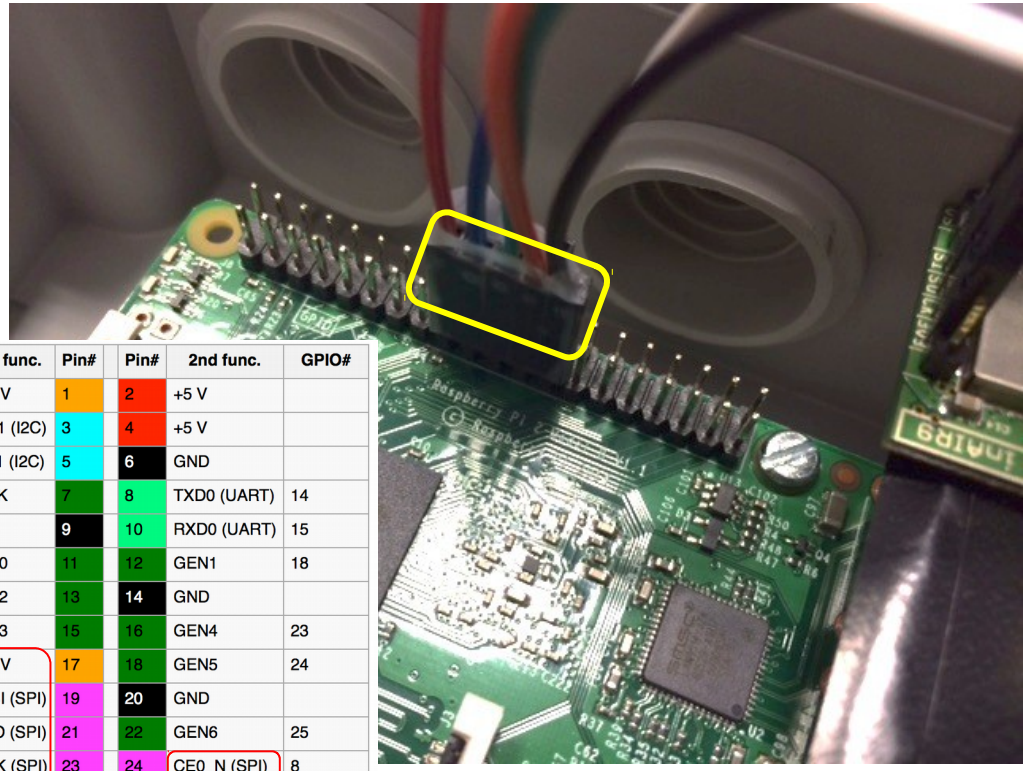


GPIO#	2nd func.	Pin#	Pin#	2nd func.	GPIO#
	+3.3 V	1	2	+5 V	
2	SDA1 (I2C)	3	4	+5 V	
3	SCL1 (I2C)	5	6	GND	
4	GCLK	7	8	TXD0 (UART)	14
	GND	9	10	RXD0 (UART)	15
17	GEN0	11	12	GEN1	18
27	GEN2	13	14	GND	
22	GEN3	15	16	GEN4	23
	+3.3 V	17	18	GEN5	24
10	MOSI (SPI)	19	20	GND	
9	MISO (SPI)	21	22	GEN6	25
11	SCLK (SPI)	23	24	CE0_N (SPI)	8
	GND	25	26	CE1_N (SPI)	7
(RPi 1 Models A and B stop here)					
EEPROM	ID_SD	27	28	ID_SC	EEPROM
5	N/A	29	30	GND	
6	N/A	31	32		12
13	N/A	33	34	GND	
19	N/A	35	36	N/A	16
26	N/A	37	38	Digital IN	20
	GND	39	40	Digital OUT	21

Depending on the model, you can have the « short » or the « long » GPIO interface. However, the SPI pins are at the same location therefore it does not change the way you connect the radio module if you take pin 1 as the reference. Connect the SPI pins (MOSI, MISO, CLK, CS) of the radio to the corresponding pins on the RPI. Note that CS goes to CE0_N on the RPI.

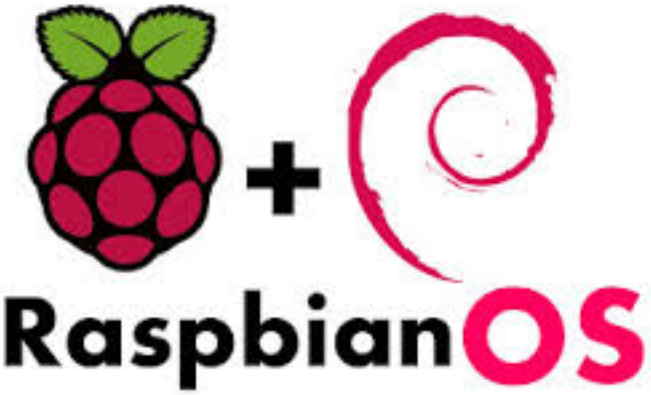
CONNECTING THE RADIO MODULE (2)

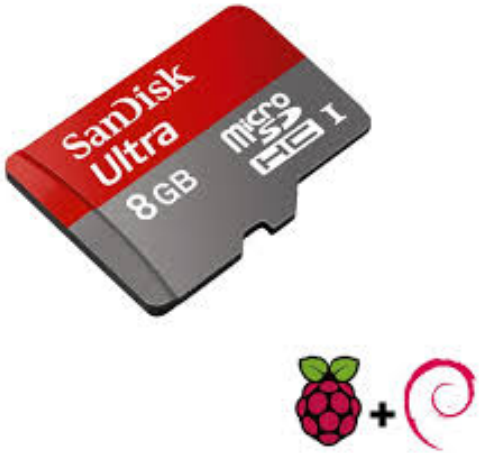




GPIO#	2nd func.	Pin#	Pin#	2nd func.	GPIO#
	+3.3 V	1	2	+5 V	
2	SDA1 (I2C)	3	4	+5 V	
3	SCL1 (I2C)	5	6	GND	
4	GCLK	7	8	TXD0 (UART)	14
	GND	9	10	RXD0 (UART)	15
17	GEN0	11	12	GEN1	18
27	GEN2	13	14	GND	
22	GEN3	15	16	GEN4	23
	+3.3 V	17	18	GEN5	24
10	MOSI (SPI)	19	20	GND	
9	MISO (SPI)	21	22	GEN6	25
11	SCLK (SPI)	23	24	CE0_N (SPI)	8
	GND	25	26	CE1_N (SPI)	7
(RPi 1 Models A and B stop here)					
EEPROM	ID_SD	27	28	ID_SC	EEPROM
5	N/A	29	30	GND	
6	N/A	31	32		12
13	N/A	33	34	GND	
19	N/A	35	36	N/A	16
26	N/A	37	38	Digital IN	20
	GND	39	40	Digital OUT	21

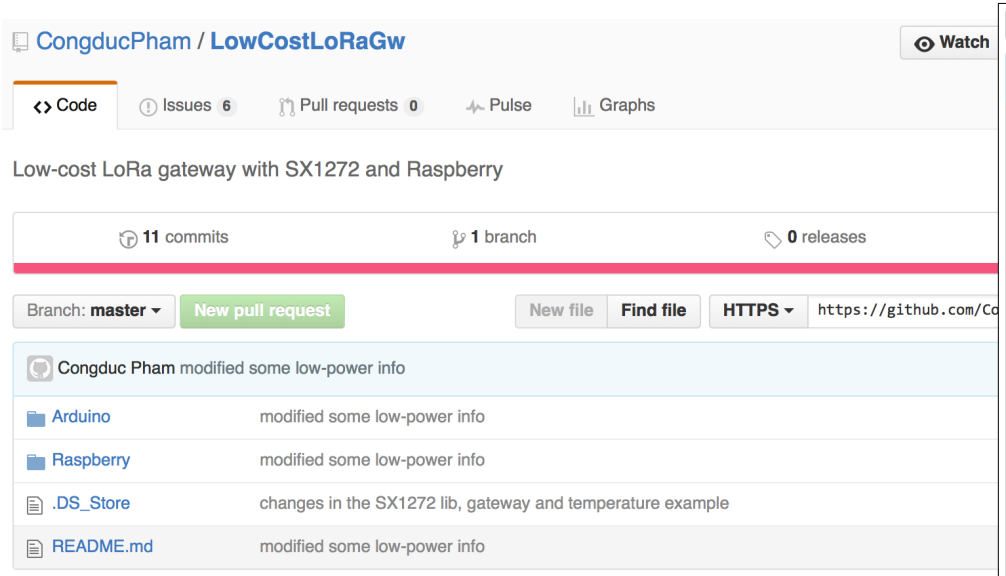
INSTALLING THE OS

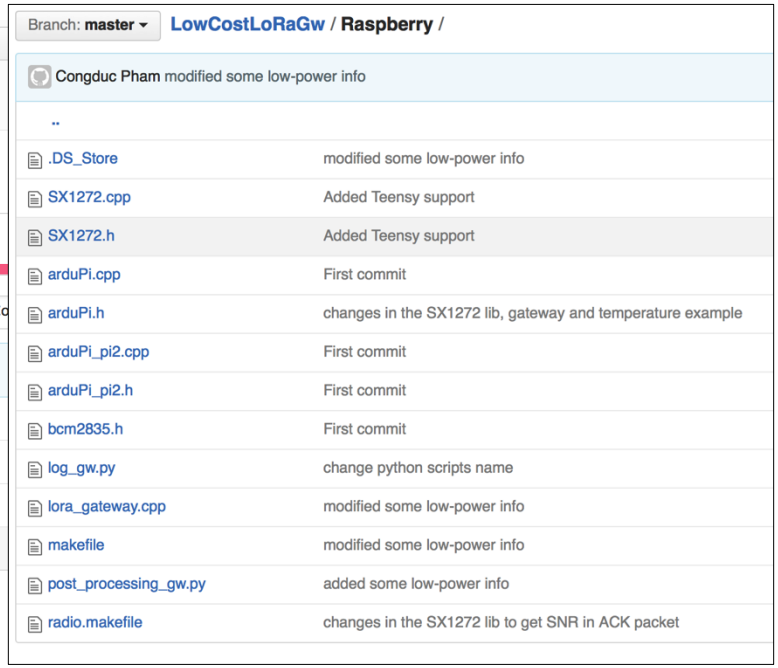




We use the Raspbian OS. Install it on an SD card. There are many tutorials on the Internet for such procedure. Alternatively, we can provide the full image to burn on the SD card. It’s 8GB!

GETTING THE LORA SPECIFIC GATEWAY SOFTWARE

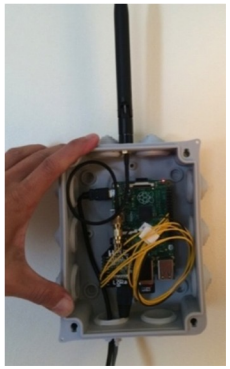


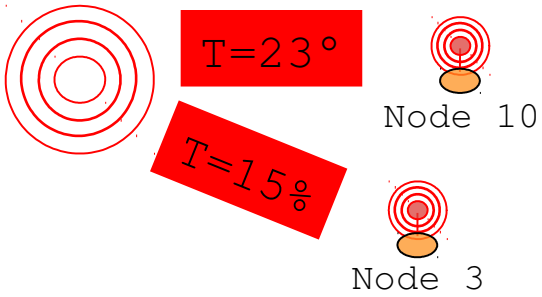



```
> mkdir lora_gateway
> git clone https://github.com/CongducPham/LowCostLoRaGw.git
> cp LowCostLoRaGw/Raspberry/* lora_gateway/
```

Log in the RPI (ssh) and create a directory called lora_gateway. Get the LoRa RPI library from our github: **https://github.com/CongducPham/LowCostLoRaGw** (right) then copy all the files of the github's Raspberry folder into the lora_gateway folder.

SATRTING THE BASIC GATEWAY






```
> sudo ./lora_gateway
Power ON: state 0
LoRa mode: 1
Setting mode: state 0
Channel CH_10_868: state 0
Power M: state 0
Get Preamble Length: state 0
Preamble Length: 8
LoRa addr 1 : state 0
SX1272/76 configured as LR-BS. Waiting RF input for transparent RF-serial bridge

--- rxlora. dst=1 type=0x10 src=10 seq=0 len=5 SNR=9 RSSIpkt=-54
^p1,16,10,0,5,9,-54
T=23°
--- rxlora. dst=1 type=0x10 src=3 seq=0 len=5 SNR=8 RSSIpkt=-54
^p1,16,3,0,5,8,-54
T=15%
```

POST-PROCESSING RECEIVED DATA

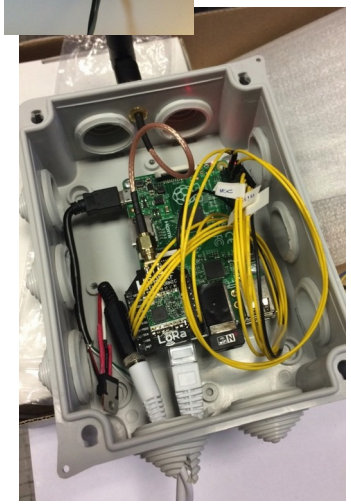
```
> sudo ./lora_gateway | python ./post_processing_gw.py
Power ON: state 0
LoRa mode: 4
Setting mode: state 0
Channel CH_10_868: state 0
Power M: state 0
Get Preamble Length: state 0
Preamble Length: 8
LoRa addr 1 : state 0
SX1272/76 configured as LR-BS. Waiting RF input for transparent RF-serial bridge
--- rxlora. dst=1 type=0x10 src=10 seq=0 len=5 SNR=9 RSSIpkt=-54
^p1,16,10,0,5,9,-54
Rcv ctrl packet info 1,16,10,0,5,9,-54
(dst=1 type=0x10 src=10 seq=0 len=5 SNR=9 RSSI=-5
T=23°
--- rxlora. dst=1 type=0x10 src=3 seq=0 len=5 SNR
^p1,16,3,0,5,8,-54
Rcv ctrl packet info 1,16,3,0,5,8,-54
(dst=1 type=0x10 src=3 seq=0 len=5 SNR=8 RSSI=-54)
H=85%
```

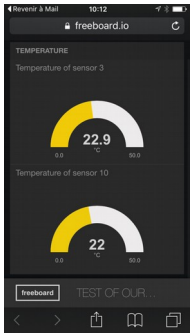
All lines that are not prefixed by specific character sequence are displayed unchanged

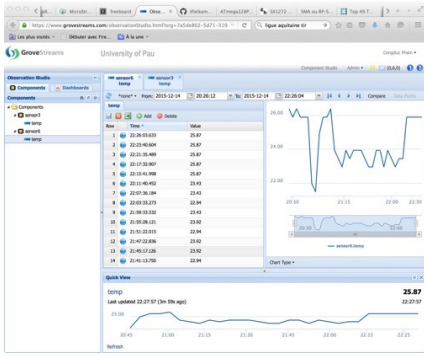
^p provides information on the last received packet: dst, type, src, seq, len, SNR & RSSI

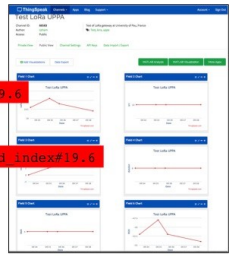
Pre-defined sequences inserted by the gateway or the end-device allow for information exchanged between the gateway and the post-processing program

FROM GATEWAY TO CLOUD



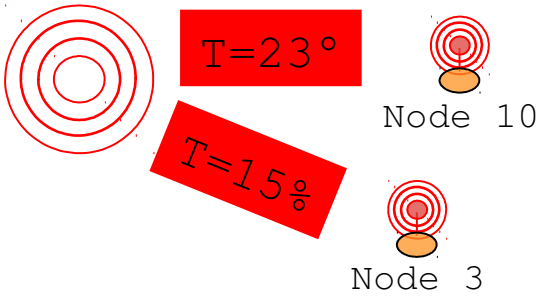






Data received at the gateway can be pushed to IoT clouds. We provide python script examples for many IoT cloud platforms. Most of clouds with REST API can be easily integrated.

LOG RECEIVED MESSAGES USING CLOUD SERVICES

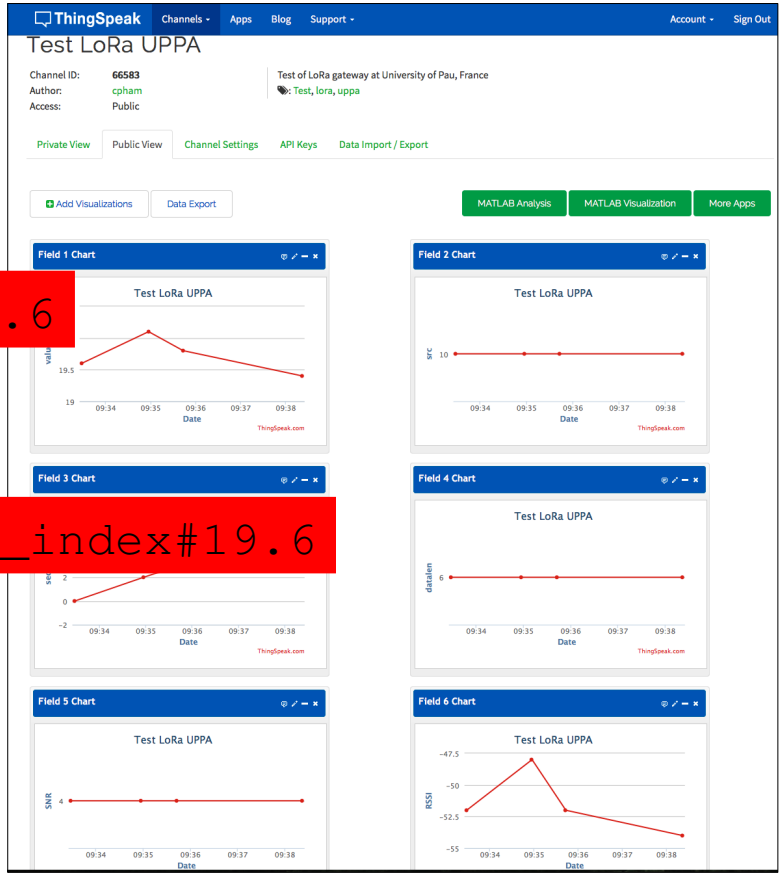



```
> sudo ./lora_gateway | python ./post_processing_gw.py
Power ON: state 0
LoRa mode: 4
Setting mode: state 0
Channel CH_10_868: state 0
Power M: state 0
Get Preamble Length: state 0
Preamble Length: 8
LoRa addr 1 : state 0
SX1272/76 configured as LR-BS. Waiting RF input for transparent RF-serial bridge

--- rxlora. dst=1 type=0x10 src=10 seq=0 len=5 SNR=9 RSSIpkt=-54
Rcv ctrl packet info 1,16,10,0,5,9,-54
(dst=1 type=0x10 src=10 seq=0 len=5 SNR=9 RSSI=-54)
rcv msg to log (\$) on dropbox : T=23°
--- rxlora. dst=1 type=0x10 src=3 seq=0 len=5 SNR=8 RSSIpkt=-54
Rcv ctrl packet info 1,16,3,0,5,8,-54
(dst=1 type=0x10 src=3 seq=0 len=5 SNR=8 RSSI=-54)
rcv msg to log (\&) on firebase : T=15%
```

\$ or \& before the data indicates that the data should be logged on a file or server. It is up to the end-device to decide which option

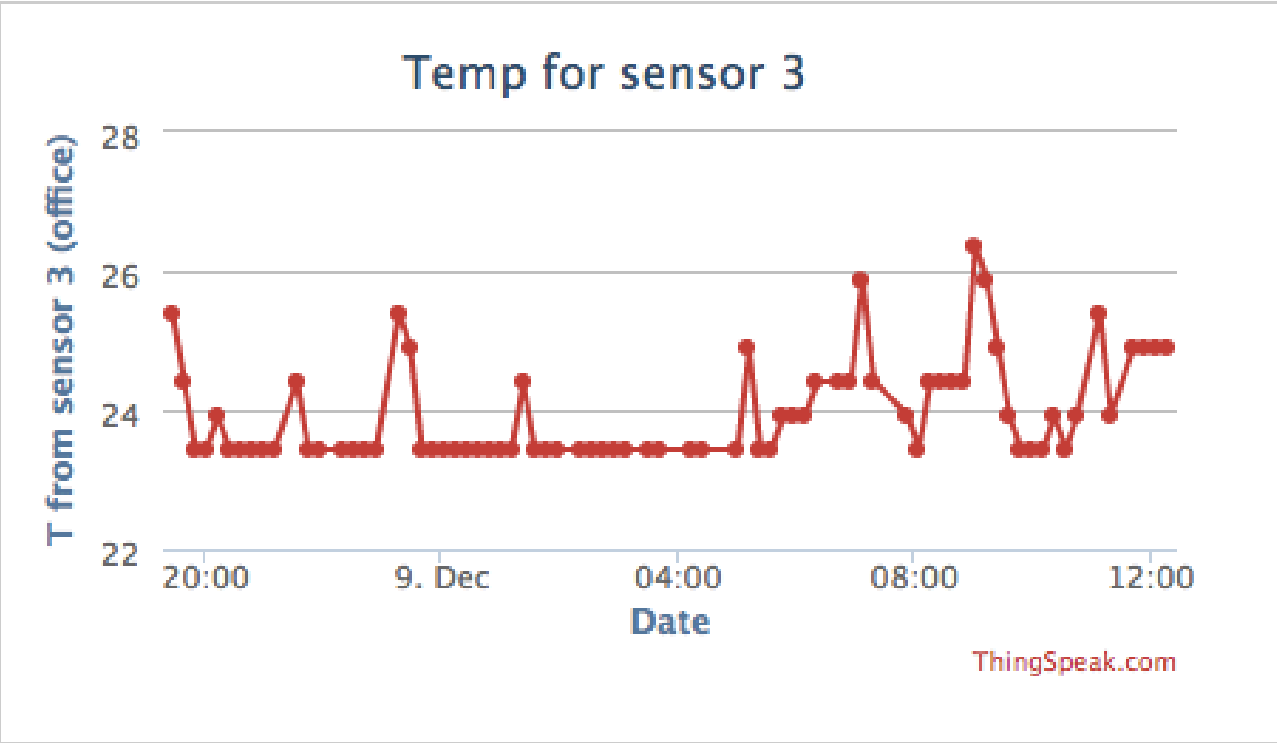
USING THINGSPEAK (1)

A message starting with '\!' is logged in a Thingspeak channel



USING THINGSPEAK (2)

> sudo ./lora_gateway | python ./post_processing_gw.py -t



RUNNING WITHOUT INTERNET ACCESS (1)

A

10-15kms

GSM GPRS

3G

Wi Fi

Internet access

LoRa

No subscription
Deploy own network
Low energy consumption

B

Wi Fi

RUNNING WITHOUT INTERNET ACCESS (2)